What is claimed is:

1. A method for implementing (Simple Object Access Protocol) SOAP-based Web services via a programming language in a computing system, comprising:

in connection with code that implements at least one SOAP-based Web service, declaring at least one SOAP handling mechanism corresponding to at least one SOAP-based Web service via a construct of said programming language;

when compiling said code, communicating with a provider object; and

generating at least one of additional code and data for use at run-time when at least one of sending and receiving a SOAP message for said at least one SOAP-based Web service occurs.

2. A method according to claim 1, wherein said declaring includes declaring at least one SOAP handling mechanism corresponding to at least one SOAP-based Web service via a C++ construct;

3. A method according to claim 2, wherein said declaring includes declaring at least one C++ attribute corresponding to at least one SOAP-based Web service.

4. A method according to claim 1, wherein when sending a SOAP message for said at least one SOAP-based Web service, the method further includes utilizing said at least one of additional code and data at run-time to generate the SOAP message.

5. A method according to claim 1, wherein when receiving a SOAP message for said at least one SOAP-based Web service, the method further includes utilizing said at least one of additional code and data at run-time to convert said SOAP message to an object of the programming language.

6. A method according to claim 1, wherein said declaring reduces the amount of code a developer writes relative to writing the code without said declaring.

7. A method according to claim 1, wherein said declaring includes describing at least

one Web service interface using embedded interface definition language (IDL).

8. A method according to claim 1, wherein said provider object communicates with the compiler of said code to generate said at least one of additional code and data.

9. A method according to claim 1, wherein via said declaring, said method hides from the developer the underlying details regarding the SOAP protocol, dispatching to the appropriate object and function, marshaling the (extensible Markup Language) XML, un-marshaling the XML, and generating the SOAP response.

10. A method according to claim 1, wherein the underlying XML packaging of the SOAP message is transported according to said at least one of sending and receiving via at least one of hypertext transfer protocol (HTTP), file transfer protocol (FTP), transmission control protocol (TCP), user datagram protocol (UDP), internet relay chat (IRC), telnet protocol and Gopher protocol.

11. A method according to claim 1, wherein said provider object is an attribute provider object and said attribute provider object has access to attribute type and marshaling information.

12. A method according to claim 1, wherein for each method of said code that is to be exposed as part of the at least one Web service, the method is introduced via an interface as part of an interface definition language (IDL) description.

13. A method according to claim 12, wherein said declaring further includes specifying at least one of in, out, size_is and retval IDL attributes in the at least one Web service's interface declaration.

14. A method according to claim 1, wherein said declaring includes declaring at least one of a soap_handler attribute, a soap_method attribute and a soap_header attribute.

15. A method according to claim 14, wherein said at least one of a soap_handler attribute, a soap_method attribute and a soap_header attribute are declared in the at least one Web service's class declaration.

16. A computer readable medium bearing computer executable instructions for carrying out the method of claim 1.

17. A modulated data signal carrying computer executable instructions for performing the method of claim 1.

18. A computing device comprising means for performing the method of claim 1.

19. A method of compiling programming language code with a compiler, comprising:

initially parsing the code;

during said initial parsing, encountering an attribute block and allowing an interface definition that follows the attribute block to include embedded information ; and

parsing the interface definition.

20. A method according to claim 19, wherein said embedded information is implemented via interface definition language (IDL).

21. A method according to claim 19, further including determining whether said attribute block includes an object attribute.

22. A method according to claim 21, wherein for each function in the interface definition, the compiler parses the function declaration, and saves the parameter attribute information for use later in determining a (Simple Object Access Protocol) SOAP message structure.

23. A method according to claim 19, wherein when the compiler encounters a soap_handler attribute, the compiler recognizes soap_handler as an attribute that is supported by an attribute provider and calls into the attribute provider.

24. A method according to claim 23, wherein the attribute provider marks the class on which the soap_handler attribute appears as handling (Simple Object Access Protocol) SOAP messages and then the attribute provider returns control to the compiler.

25. A method according to clam 24, wherein after said return of control, the compiler begins parsing the body of the class on which the soap_handler attribute appears and parses each function in the class.

26. A method according to 25, wherein for each soap_header attribute on the function, the compiler performs soap_header attribute processing.

27. A method according to claim 26, wherein said soap_header attribute processing includes recognizing a soap_header attribute as an attribute that is supported by the attribute provider and calling into the attribute provider whereby the attribute provider (i) gathers information to create the "Header" part of a SOAP message, (ii) queries the compiler for information about the parameters to the soap_header attribute and (iii) queries the compiler for information about the data type for the header

28. A method according to claim 26, wherein said soap_header attribute processing includes generating the information for creating a portion of the (Simple Object Access Protocol) SOAP message header and returning control from the attribute provider to the compiler.

29. A method according to 25, wherein if the function has a soap_method attribute, the compiler performs soap_method attribute processing

30. A method according to claim 29, wherein said soap_method attribute processing includes recognizing a soap_method attribute as an attribute that is supported by the attribute provider and calls into the attribute provider whereby the attribute provider (i) gathers information to create a (Simple Object Access Protocol) SOAP message for the function and

(ii) queries the compiler for information about the parameters to the function on which soap_method attribute appears.

31. A method according to claim 30, wherein for each parameter on the function, the attribute provider (i) uses the interface definition information to determine which function parameters are to be received as part of the incoming SOAP message, and which parameters are to be sent back as part of the SOAP message response and (ii) queries the compiler about the data type of the parameter.

32. A method according to claim 29, wherein said soap_method attribute processing includes generating the data to create the SOAP message portion for a parameter and returning control from the attribute provider to the compiler.

33. A computer readable medium bearing computer executable instructions for carrying out the method of claim 19.

34. A modulated data signal carrying computer executable instructions for performing the method of claim 19.

35. A computing device comprising means for performing the method of claim 19.

36. A method processing (Simple Object Access Protocol) SOAP messages at run-time in a computing system having a server with at least one application having at least one user object implementing at least one SOAP-based Web service, comprising:

at least one of receiving an incoming SOAP message from a client computing device and receiving a call from a user object with at least one programming language construct; and

with a runtime intermediate layer, at least one of converting said incoming SOAP message to a corresponding programming language construct and converting said call with at least one programming language construct to an outgoing SOAP message,

wherein said runtime intermediate layer utilizes at least one of code and data generated during compilation of said user object.

37. A method according to claim 36, wherein said programming language is C++.

38. A method according to claim 36, wherein said converting of said incoming SOAP message to a corresponding programming language construct includes determining the appropriate user object to receive the SOAP message.

39. A method according to claim 36, wherein said converting said call with at least one programming language construct to an outgoing SOAP message includes formatting the message for delivery to an intended client computing device.

40. A computer readable medium bearing computer executable instructions for carrying out the method of claim 36.

41. A modulated data signal carrying computer executable instructions for performing the method of claim 36.

42. A computing device comprising means for performing the method of claim 36.